ISO TC184/SC4/QC N027

Date: 1997-04-29

Supersedes SC4/QC/N019

ISO/WD 10303
Procedures for application interpretation

ABSTRACT:

This document contains the procedures for the interpretation of ISO 10303 application protocols. This document is intended to provide sufficient information that a trained interpretation resource would be able to use the guidelines in this document to produce a quality interpretation. The document includes an overview of the interpretation process, including examples; fundamental concepts of key ISO 10303 integrated resources; details to aid understanding of the decision making process at detailed levels of the interpretation process; as well as application interpreted model quality metrics.

KEYWORDS:

application interpreted model, application protocol, integrated resource, interpretation

COMMENTS TO READER:

The objective of this document is to provide procedures for the interpretation process such that the reader is able to begin to understand and perform application interpretation. The reader is expected to be familiar with EXPRESS, the ISO 10303 integrated resources, ISO 10303-13 and the SC4 Standing Documents including *Guidelines for the development and approval of ISO 10303 application protocols*, *Guidelines for application interpreted model development*, *Guidelines for the development of mapping tables*, and *Guidelines for AIC development*. The initial content for this document was developed during a two week interpretation workshop for AP 216; the content is to be increased in subsequent versions.

Project Leader:	Allison Barnard Feeney	**Project Editor:**	Mark Palmer
Address:	National Institute of Standards and Technology Building 220, Room A127 Gaithersburg, MD 20899 USA	**Address:**	National Institute of Standards and Technology Building 226, Room B306 Gaithersburg, MD 20899 USA
Telephone:	+1 301 975-3181	**Telephone:**	+1 301 975-5858
Telefacsimile:	+1 301 258-9749	**Telefacsimile:**	+1 301 975-5433
Electronic mail:	abf@nist.gov	**Electronic mail:**	mark.palmer@nist.gov

Procedures for application interpretation

29 April 1997

Allison Barnard Feeney - NIST
Jesse Crusey - NIST
Julian Fowler - PDT Solutions
Mitch Gilbert - PDIT
Jochen Haenisch - Det Norske Veritas
Rob Howard - Lloyd’s Register of Shipping
Yuanxie Janke-Zhao - KCS Consulting
Pete Lazo - Newport News Shipbuilding
Mark E. Palmer - NIST
Greg Paul - Lockheed
David Price - IBM
Anne Wasmer - DiK Darmstadt
Yuhwei Yang - PDIT

Contents

Page

Annexes

Figures

Tables

Foreword

The International Organization for Standardization (ISO) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

This standing document was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

ISO/TC 184/SC4 standards are prepared according to guidelines put forth in the following standing documents:

— Guidelines for application interpreted construct development;

— Guidelines for application interpreted model development;

— Guidelines for the development and approval of STEP application protocols;

— Guidelines for the development of abstract test suites;

— Guidelines for the development of mapping tables;

— ISO/TC 184/SC4 organization handbook;

— STEP architecture and methology and reference manual;

— Supplementary directives for the drafting and presentation of ISO 10303.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application interpreted constructs, application protocols, abstract test suites, implementation methods, and conformance testing. The series are described in ISO 10303-1.

The purpose of this document is to provide procedures for application interpretation of ISO 10303 (commonly called STEP) application protocols. The objective is to provide sufficient detail that a trained interpretation resource would be able to use the procedures in this document to produce a quality interpretation. The expected audience for this document is, therefore, potential application interpreters as well as developers of ISO 10303 application protocols. This is one of several documents intended to be of use in the development of ISO 10303 application protocols. Others include *Guidelines for the development and approval of STEP application protocols*, *Guidelines for application interpreted model development*, *Guidelines for the development of mapping tables*, and *Guidelines for application interpreted construct development*. The *STEP architecture and methodology reference manual* provides a more high-level view of the ISO 10303 development methodology.

Interpretation is the process, mechanism, or manner by which meaning is assigned to an abstract representation of an event, object, or concept. Within an AIM, the abstract representation is specified by an EXPRESS construct. The interpretation of a construct defined in an integrated resource results in the creation of a new construct in the AIM that may restrict, narrow, or constrain the semantic scope of the integrated resource construct, thereby *specializing* the construct.

Interpretation is ultimately grounded in human understanding. Interpretation draws not only on the meaning of the constructs themselves, but the context within which the constructs are generated, used, or received. The context specified in the ARM is defined by such things as the life cycle stage and the application domain that bounds the scope of the AP. The scope of ISO 10303 is the representation of product data. The integrated resources were developed within the framework of that context; the context of the integrated resources is not limited to a specific life cycle stage or by type of product. Because the integrated resources and the ARMs are developed as representations of information in different (though established and related) contexts, they are subjected to the influence of different contextual factors. In order to account for contextual factors, interpretation of integrated resources in ISO 10303 (particularly the selection of integrated resource constructs) relies on human comprehension of the requirements represented in the ARM and the in-depth knowledge of the semantics and contextual factors under which the ISO 10303 resources are developed.

The interpretation process is a formal and established part of the AP development process. ISO 10303 has defined standardized resources that are used as the basis for interpretation: the integrated resources and AICs[1].

Individual AP projects are responsible for development and documentation of the entire application protocol, including application interpretation. The interpretation process is interactive and requires several participants playing different roles in order to reach successful completion. The AP project team brings to the process a detailed understanding of the domain information requirements and the interpretation resources bring a detailed understanding of the integrated resources and the interpretation process. An application reference model (ARM) never explicitly documents all requirements, so much of the interaction between the participants in an interpretation workshop is focused on discovering all the requirements that are both implicit and explicit in the ARM. A number of AIM development workshops are required to complete the interpretation for each application protocol project; the exact number of workshops depends on the complexity of the information requirements and the size of the scope for the application protocol's domain.

The AIM development activities, of which the interpretation process is the primary focus, are organized into four phases: preparation, interpretation, documentation, and review/sign-off. The technical activities involved in the first and third phases are described in *Guidelines for application interpreted model development*. Review and sign-off phase follows the procedures defined in the *ISO TC184/SC4 organization handbook* . Detailed technical procedures and examples for the interpretation stage are the focus of this document. The interpretation process is presented in the remainder of this document through a description of each step of the interpretation process.

[1] The term “resources” is used through out this document to refer to both constructs in the integrated resources and the AICs.

Procedures for application interpretation

1 Scope

This SC4 standing document specifies procedures for the interpretation of ISO 10303 application protocols.

The following are within scope of this standing document:

— list of prerequisites to learning to perform application interpretation;

— a listing of entities from the ISO 10303 integrated resources that are are common to all ISO 10303 application ~~protocols~~ interpreted models;

— explanation of the fundamental concepts of the ISO 10303 integrated resources;

— description of the detailed steps of the interpretation process for ISO 10303 application protocols, including examples;

— detailed guidance on how to select constructs of the ISO 10303 integrated resources that map to the information requirements of an ISO 10303 application protocol;

—metrics by which to assess the quality of each step of the application interpretation process;

— explanation of consistency checks that verify the quality of the interpretation;

The following are outside the scope of this standing document:

— specification of presentation information for the documentation of any portion of an ISO 103030 application protocol;

NOTE 1 - This information is found in *Supplementary directives for the drafting and presentation of ISO 10303*.

— decomposition of the process of developing an application interpreted model short form EXPRESS schema;

— guidelines for the use of the EXPRESS language for writing an application interpreted model for an ISO 10303 application protocol;

— EXPRESS templates for commonly used global rules;

— a list of steps for writing application interpreted model short form EXPRESS;

— tables listing string values for integrated resource entity attributes that have been standardized within ISO 10303 application protocols;

NOTE 2 - This information in the previous five bullets is covered in the document entitled *Guidelines for application interpreted model development*.

— guidelines for the use of EXPRESS in information models other than ISO 10303 application interpreted models.

NOTE 3 - The EXPRESS language is described in ISO 10303-11. This document provides EXPRESS usage guidance only in the context of application interpreted model development.

2 Normative references

The following standing documents and standards contain provisions which, through reference in this text, constitute provisions of this standing document. At the time of publication, the editions indicated were valid. All standing documents and standards are subject to revision, and parties to agreements based on this standing document are encouraged to investigate the possibility of applying the most recent editions of the standing documents and standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/TC 184/SC4 N534:1997, *Guidelines for application interpreted construct development.*

ISO/TC 184/SC4 N532:1997, *Guidelines for application interpreted model development.*

ISO/TC 184/SC4 N535:1997, *Guidelines for the development and approval of STEP application protocols.*

ISO/TC 184/SC4 N533:1997, *Guidelines for the development of mapping tables.*

ISO/TC 184/SC4 N537:1997, *Supplementary directives for the drafting and presentation of ISO 10303.*

ISO 10303-11:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: Language reference manual.*

3 Definitions and abbreviations

3.1 Terms defined in ISO 10303-1

This SC4 standing document makes use of the following terms defined in ISO 10303-1.

3.1.1 application: a group of one or more processes creating or using product data.

3.1.2 application activity model (AAM): a model that describes an application in terms of its processes and information flows.

3.1.3 application context: the environment in which the integrated resources are interpreted to support the use of product data in a specific application.

3.1.4 application interpreted model (AIM): an information model that uses the integrated resources necessary to satisfy the information requirements and constraints of an application reference model, within an application protocol.

3.1.5 application object: an atomic element of an application reference model that defines a unique concept of the application and contains attributes specifying the data elements of the object.

3.1.6 application protocol (AP): a part of this International Standard that specifies an application interpreted model satisfying the scope and information requirements for a specific application.

NOTE: This definition differs from the definition used in open system interconnection (OSI) standards. However, since this International Standard is not intended to be used directly with OSI communications, no confusion should arise.

3.1.7 application reference model (ARM): an information model that describes the information requirements and constraints of a specific application context.

3.1.8 conformance class: a subset of an application protocol for which conformance may be claimed.

3.1.9 data: a representation of information in a formal manner suitable for communication, interpretation, or processing by human beings or computers.

3.1.10 implementation method: a part of this International Standard that specifies a technique used by computer systems to exchange product data that is described using the EXPRESS data specification language [ISO 10303-11].

3.1.11 interpretation: the process of adapting a resource construct from the integrated resources to satisfy a requirement of an application protocol. This may involve the addition of restrictions on attributes, the addition of constraints, the addition of relationships among resource constructs**.**

3.1.12 product: a thing or substance produced by a natural or artificial process.

3.1.13 product data: a representation of information about a product in a formal manner suitable for communication, interpretation, or processing by human beings or by computers.

3.1.14 resource construct: a collection of EXPRESS entities, types, functions, rules and references that together define a valid description of an aspect of product data.

3.1.15 unit of functionality (UoF): a collection of application objects and their relationships that defines one or more concepts within the application context such that removal of any component would render the concepts incomplete or ambiguous.

3.2 Terms defined in ISO 10303-202

This SC4 standing document makes use of the following terms defined in ISO 10303-202.

3.2.1 application interpreted construct (AIC): a logical grouping of interpreted constructs that supports a specific function for the usage of product data across multiple application contexts.

3.3 Other definitions

For the purposes of this SC4 standing document, the following definitions apply

3.3.1 information unit: a grouping of related constructs (entity data types, attributes and relationships) that together represent one of the high-level concepts of the ISO 10303 data architecture.

3.4 Abbreviations

For the purposes of this SC4 standing document, the following abbreviations apply.

AAM application activity model

AIC application interpreted construct

AIM application interpreted model

AP application protocol

ARM application reference model

CC conformance class

IR integrated resource

UoF unit of functionality

4 Prerequisites

The baseline prerequisites for someone to learn the process of application interpretation include:

— some exposure to interpretation (either through participation as AP editor, domain expert, observer, etc.);

— exposure to multiple modeling methodologies. ARM modeling styles vary dramatically from project to project, and the ability to understand models developed in a variety of languages, from a variety of perspectives is useful;

— understanding of the ISO 10303 integrated resource parts. Such an understanding can come through study and review of the usage of the IRs as documented in completed ISO 10303 application protocols, and as documented in interpretation reports;

— understanding of the EXPRESS language. A useful understanding of EXPRESS best comes through participation in the development of an application protocol's AIM short form;

— understanding of the ISO 10303 architecure and methodology as documented in *STEP architecture and methodology reference manual* and the detailed guidance on AP development documented in the SC4 Standing Documents.

The best way to learn the interpretation process is through active participation in interpretation workshops then studying the resulting interpretation documenation. The availability of interpretation workshops in which to learn these skills is dependent on the schedules of other AP development projects.

5 Fundamental concepts of ISO 10303 integrated resources

5.1 Introduction to STEP data architecture

The STEP data architecture is represented by the information model found in the integrated resources. Integrated resources are application-context-independent product data models that provide standard constructs used in the creation of ISO 10303 application interpreted models.

The integrated resources are data models that reflect and support the common requirements of many different product data application areas. The integrated resources constitute a single conceptual model for product data. The constructs within this model are the basic semantic elements that are used for the description of any product in terms of an unlimited number of life-cycle specific views of that product. The properties that characterise a product do so indirectly, through characterisation of these views, rather than the product directly. These properties, in turn, may be represented in many different ways depending upon the nature of the view being described. The integrated resources define reusable components that are intended to be combined and refined in the application interpreted models to meet a specific need.

The basic conceptual framework described above is called the STEP data architecture, and it is the basis for all ISO 10303 data specifications. The STEP data architecture is the framework upon which all of the integrated resources are built and it is reflected in all of the application interpreted models of all application protocols. The models that capture this framework embody the principle of existence dependence which ensures that all product information in a given set of data that conforms to an ISO 10303 application protocolis related to a product and, ultimately and application context.

The principle of existence dependence is the basis for the discussion of the ISO 10303 information units in clauses 5.3 through 5.12. The entities that are at the core of the STEP data architecture and are required to be incorporated into every application interpreted model are specified in clause 5.2.

5.2 Required constructs

The following entities are an important part of the ISO 10303 data architecture. They appear in every AIM documented to date. Though they are likely to be specialized (subtyped) in AIMs, they provide the core data constructs for the AIM.

ISO 10303-41:1994	`application_context`
ISO 10303-41:1994	`application_context_element`
ISO 10303-41:1994	`product`
ISO 10303-41:1994	`product_context`
ISO 10303-41:1994	`product_definition_context`
ISO 10303-41:1994	`product_definition`
ISO 10303-41:1994	`product_definition_formation`

For APs with requirements that specify technical or scientific data about properties of products, the following are the core property data constructs for the AIM.

ISO 10303-41:1994	`property_definition`
ISO 10303-41:1994	`property_definition_representation`
ISO 10303-41:1994	`shape_definition_representation`
ISO 10303-43:1994	`representation`
ISO 10303-41:1994	`product_definition_shape`
ISO 10303-41:1994	`shape_representation`
ISO 10303-43:1994	`representation_item`

5.3 Basic information units ~~Fundamental concepts~~

~~It is recognized that there is insufficient information available, either within the standardized integrated resources parts of ISO 10303, or within existing "methods" documents, on the basic "information units" of ISO 10303. This clause of this document addresses this issue.~~

5.3.1 Introduction

The modeling principle of existence dependency that is used within the integrated resources provides the capablity to manage the product data constructs at a level higher than the entity. The use of certain entities within the integrated resources implies the use of other entities. This fact provides the basis for analyzing information requirements within the context of the use of information units within the integrated resources in application interpreted models. There are a few basic ideas within the ISO 10303 architecture and methodology that form the foundation for this type of requirements analysis.

The following ideas form the basis of the ISO 10303 architecture and methodology:

— the ISO 10303 integrated resources comprise a single model for product data;

— this model is based on an underlying framework - the "Generic Product Data Model" (GPDM); and

— the model supports product data applications (application protocols) through different, specific usages of generic constructs.

~~Unfortunately, these concepts are not articulated clearly in the documentation of the ISO 10303 standards, and are therefore not widely understood. Much effort has been expended over the years in providing documentation that is intended to address this need: however, it is clear that little of this documentation has really reached its intended audience.~~

One of the associated concepts is that the (EXPRESS) entity data type is not always a fundamental unit of definition for the ISO 10303 integrated resources. Rather, the foundation of the integrated resources is a series of inter-related and nested "information units" that form the core of the Generic Product Data Resources (GPDR), that are primarily documented in ISO 10303-41. The main information units are discussed in this document, and are illustrated by the diagram below.

[Diagram: ISO 10303 information units]

Figure 1 - "Information units" of the ISO 10303 integrated resources

In figure 1, the arrows should be read as "contains", so:

— the product category information unit contains the product information unit;

— the representation information unit contains the product property AND product definition AND product information units.

This structure results from the ~~desric~~principle that any information exchanged using any ISO 10303 application protocol should be directly and explicitly related to both the product(s) to which it pertains and to the context(s) for which it is valid.

5.3.2 Product and product context

As a standard for product data representation, the focal point of the ISO 10303 integrated resources is the product entity data type. This data type serves two purposes:

— to identify a product as being of interest, and give it a name and a description;

— to characterize the context in which the product and its associated data exist and are valid.

The entity types product and product_context (which is a subtype of application_context_element and is therefore dependent on application_context) provide application protocols with the capability to hold this information. This "information_unit" is illustrated in figure 2.

[EXPRESS-G diagram: product]

Figure 2 - product information unit

The definition of "product" as given in part 1 of ISO 10303 is repeated in 3.1.12 of this document. This definition needs to be distinguished from the definition of the product entity data type, which provides the identification of products and characterization of their context, as is stated in the definition from ISO 10303-41:

“product: a product is the identification and description, in an application context, of a physically realizable object that is produced by a process.”

The usage of the application_context_schema is discussed in 5.11.

5.3.3 Identifying configurations for products

This information unit supports requirements for identification, description and inter-relationships amongst different versions, variants, or configurations for products. As with the product information unit, it is important to recognize that the information held here is still at a very simple level: just as the product simply holds an identifier and a relationship to a context for a product, so product_definition_formation holds an identifier and a relationship to the product of which it is a version. The constructs of this information unit are illustrated in figure 3.

[EXPRESS-G diagram: product_definition_formation]

Figure 3 - The product configuration information unit

Note: "non-standard" EXPRESS-G is used in this document: an entity data type whose representation is a box with a heavy outline is taken to represent the "information unit" that includes the data type. Thus the product configuration information unit includes, as part of its definition, all the constructs that define the product information unit.

5.3.4 Three ways of thinking about products

An understanding of the detailed structure and semantics of the ISO 10303 integrated resources is based on their "data architecture" - a high level structure that has been used to create and maintain the syntactic and semantic consistency of the model. (It is worth pointing out here that, in spite of its publication in multi-document, multi-schema form, the integrated resources are a single conceptual model for use in product data applications.)

The first aspect of the ISO 10303 data architecture is that it recognized three distinct, high level "views" or potential uses of product data. These are illustrated in figure 4. This depicts the way in which, for a given product, there are orthogonal concepts related to:

— how the product is classified or categorized;

— how the product is presented to the market;

— the technical description of the product for the purpose of design, engineering, manufacturing, operations, maintenance, etc.

Each of these concepts is supported by a major information unit within the ISO 10303 integrated resources.

[Three views of product]

Figure 4 - Three ways of thinking about products

5.3.5 Classification or categorization of products

One of the common requirements of many different application domains is to be able to classify or categorize products. This requirement is supported by the ISO 10303 integrated resources illustrated in figure 5.

[EXPRESS-G diagram: product_category]

Figure 5 - The product category information unit

The constructs within this information allow different applications to use common, generic data model elements for the following requirements:

— identifying and describing classes/categories of products (product_category);

— establishing relationships between different classes of product (product_category_relationship; a typical usage of this construct is to describe subclass/superclass relationships;

— identifying the product(s) (instances of the product entity data type) that are classified as being members of a class (product_related_product_category).

This information unit therefore provides a classification view of products.

5.3.6 Presenting products to the market

Products are not just designed and manufactured: they are also sold and marketed to (potential) customers. This is represented in the ISO 10303 integrated resources by an information unit that focuses on the "marketing" view of products, and the relationship of this information to the technical description (design) of the products concerned. The major constructs of this information unit are illustrated in figure 6.

[EXPRESS-G diagram: product_concept]

Figure 6 - The product_concept information unit

This information unit may be used in application protocols to exchange information about the way in which a product is identified and presented to the market. The "central" entity data type is product_concept: this identifies, names and describes, for a given market (product_concept_context), something that is (or is to be) sold and supplied. One possible usage of this construct is to identify and hold information about a particular make and model of automobile or aeroplane.

The remaining constructs in this information unit provide the capability to identify and describe different items (configuration_item) that fulfil the requirements of a product concept, and to associate (configuration_design) these items with the technical information for their design, etc.
This information unit therefore provides the marketing view of products.

5.3.7 Differentiating different life-cycle or discipline “views”

The third "axis" of the ISO 10303 data architecture comprises the information units that hold all the scientific, engineering, and other technical information that is created and used throughout the product life cycle. Any engineered product will obviously have a great deal of information related to it: however, for the purposes of exchanging data it is rarely (if ever) required that all this information is exchanged at the same time. Different information will be generated and used in different life cycle phases, and by the different disciplines that contribute to the definition of the product.

The ISO 10303 integrated resources provide a capability that allows application protocols to specify how technical information is aggregated for these different life-cycle and/or discipline purposes. This product definition information unit is illustrated below.

[EXPRESS-G diagram: product_definition]

Figure 7 - The product definition information unit

The product_definition entity data type is used in application protocols to denote a definition of a product that pertains to a particular life-cycle phase (product_definition_context). Product definitions are then used as the "aggregator" for properties and other information that is valid for that context. Product definitions may be associated with other product definitions (product_definition_relationship). This latter entity data type is the basis for ISO 10303's capabilities to describe the composition of products, assembly structures, connectivity.

5.3.8 Identifying properties and relating them to views

As noted above, the product definition information unit provides application protocols with the capability to identify, describe and inter-relate definitions that are used to aggregate the properties that pertain to different life-cycle phases and/or disciplines. These properties must, of course, themselves be defined. Within the ISO 10303 integrated resources properties are defined such that they are dependent on elements within the product definition information unit (i.e., a property is always a property "of" something. Properties are, however, independent of the data values that may be used to describe or represent the property. This means that within an application protocol a property (such as shape or material) may be identified for a product, and that zero, one or many collections of data that represent this property may be described separately.

This information unit is illustrated below.

[EXPRESS-G diagram: property_definition]

Figure 8 - The property definition information unit

NOTES:
1 - Again, "non-standard" EXPRESS-G is used here: the broken relationship between property_definition and characterized_definition indicates that the actual resource model contains additional constructs here. This detail is unnecessary for the high-level discussion here.

2 - The property_definition_relationship entity data type is defined in ISO 10303-45.

The constructs shown here allow properties to be identified and described (property_definition) for the product_definition or product_definition_relationship that possesses the property. They also allow property definitions to be associated with other property definitions (property_definition_relationship).

Other elements of the ISO 10303 integrated resources specialize the constructs in this information unit for particular types of property. For example:

— the product_property_definition_schema (ISO 10303-41) includes the constructs that are used to identify shape properties;

— the material_property_definition_schema (ISO 10303-45) includes the constructs that are used to identify material properties .

5.3.9 Collecting data for properties

The final core information unit of the ISO 10303 integrated resources provides application protocols with the capability to create coherent collections of data items and to relate these collections to property definitions. The constructs within this information unit are illustrated in figure 9.

[EXPRESS-G diagram: representation]

Figure 9 - The representation information unit

This information unit allows application protocols to collect data elements (representation_item). The collection is a representation, which also provides the frame of reference (representation_context) that makes it possible to understand the relationships between items. Many of the more domain or discipline-specific integrated resources are concerned with the specification of types of representation_item. Sometimes representation_items may be purely descriptive (textual), sometimes they may be name-value pairs, whilst some are complex data structures of themselves (geometry, presentation).

Collections of data items may be associated (representation_relationship) with other collections; through a mechanism not shown here (the mapped_item construct defined in ISO 10303-43) one collection may be used as part of the definition of another.

Each representation may be associated (property_definition_representation) with one or more property definitions. This is a "many-to-many association. For example:

— a behavioral property may be represented by an equation, or by tabular data values, or by the shape of a graph

— one set of tabular data values may be used to represent many different behavioral properties (which may be the properties of the same product or of different products)

5.4 Usage of the application_context schema

5.4.1 Principles of the application_context schema

The application_context_schema (defined in ISO 10303-41) is the basis for capturing "meta-data" for a given application protocol, i.e., it provides a number of entity data types whose populations will qualify or constrain the applicability of product data. This section expands on the definitions of this schema given in part 41, and provides a basis of a "usage guide" for AP developers, implementors, and end-users.

The application_context_schema is illustrated in figure 10.

[EXPRESS-G diagram: application_context_schema]

Figure 10 - application_context schema

[Additional material should be taken from Bill Danner's paper]

5.4.2 An example using a shipbuilding application

~~The application_context_schema (defined in ISO 10303-41) is the basis for capturing "meta-data" for a given application protocol, i.e., it provides a number of entity data types whose populations will qualify or constrain the applicability of product data. This section expands on the definitions of this schema given in part 41, and provides a basis of a "usage guide" for AP developers, implementors, and end-users.~~

The use of the application_context_schema is illustrated by the two scenarios illustrated below, taken from AP216.

[IDEF0 diagram: scenario A]

Figure 11 - Scenario A

In Scenario "A" (above), naval architects within a shipyard produce a shape model of a hull moulded form, within a design ship activity. This shape model is exchanged with a model basin, where naval architects undertake the analyze design activity and provide feedback.

[IDEF0 diagram: scenario B]

Figure 12 - Scenario B

In Scenario "B" (above), naval architects within a shipyard produce a shape model of a hull moulded form, as before. This shape model is in this case exchanged with a classification society, which undertakes the approve design activity and provide feedback.

Table 1 below illustrates a possible population of the entity data types of the application_context_schema for these two scenarios.

Table 1 - Possible population of entity data tupes for scenarios A and B

Attribute	Scenario "A"	Scenario "B"
product.frame_of_reference -> product_context <= application_context_element.name	'naval architecture'	'naval architecture'
product.frame_of_reference -> product_context.discipline_type	'ship'	'ship'

Attribute	Scenario "A"	Scenario "B"
product.frame_of_reference -> product_context <= application_context_element.frame_of_reference -> application_context.application	'ship building'	'ship classification'
product_definition.frame_of_reference -> product_definition_context <= application_context_element.name	‘design’	‘design’
product_definition.frame_of_reference -> product_definition_context.life_cycle_stage	‘physical design’	‘physical design’
product_definition.frame_of_reference -> product_definition_context <= application_context_element.frame_of_reference -> application_context.application	‘hull fairing’	‘hull fairing’

It was agreed at the interpretation guidelines workshop that:

— a role of the SC4 Quality Committee shall be to maintain a dictionary of all standardized values for application_context_schema attributes and their usages; and

— each application protocol project should perform a consistency check to ensure that the standardized values for application_context_schema attributes fit with the usage scenarios described for the application protocol.

~~5.5 Summary~~

~~This clause has described the major "information units" that form the basis of the ISO 10303 integrated resources. Understanding of the integrated resources as a whole, their usage in application protocols, and the discipline of application interpretation depends on a thorough understanding of these underlying structures.~~

6 Interpretation process overview

This clause describes the detailed steps of the interpretation process. The interpretation process is decomposed into three phases. The steps of each phase are described, along with metrics for assessing the quality of the execution of that step and examples that illustrate the step.

6.1 Phase 1: Preparation

6.1.1 Step 1: Interpretation expert preparation

Process: Interpretation experts ~~must~~ review and study ARM and information requirements prior to the interpretation workshop. Determine the context of the AP: what is th~~e context of the~~application's viewpoint ~~of~~on the product? The context of the AP ~~should~~will be ~~stated~~ captured in the Product_context. Whenever a mapping ~~you map something~~ to product_definition is being considered, the life cycle stage of the product is considered and documented as a mapping rule for that particular mapping.~~you want to know what the lifecycle stage is and document that in that mapping.~~ There are two mandatory contexts in every AP, product_context and product_definition_context. In preparing for the interpretation, the interpretation experts analyze the different products within the scope of the AP, and the life cycle views of these products.

Metric: Reveiwers comfortable with their understanding of the context, or have identified questions that will help them reach the understanding.

Example: The context of AP 216 is ship. One of the basic concepts that follows AP 216 is it's participation in the suite of shipbuilding APs. In the process of interpreting AP 216 ~~we are also trying to set~~ the context of the suite of APs is also a major consideration. ~~application_context_element.name for the product. For the product_definition the context is the lifecycle stage. Should be set to design in the case of AP 216.~~ Since the shipbuilding context is that of ship in general as well, the product_context.discipline_type attribute will be constrained to be a fixed value of 'ship'within the AP. After discussions with the AP team the life cycle stage for all products in the AP is the physical design of the ship. Therefore, within AP216, the only legal value for product_definition_context.life_cycle_stage will be 'physical design'. This value will be constrained within the AIM. (See table 1 in 5.10 for example with typical values for population of the .name, .application and .lifecycle_stage attributes.)

6.1.2 Step 2: AP team preparation

Process: Domain experts must evaluate what materials will be required to best explain the requirements of the application domain. This could include ARM populations, industrial data (drawings, tables, specifications, etc.) Distribute document prior to workshop. A well-documented and understood ARM is the starting point for application interpretation.

Metric: Required documentation distributed prior to meeting. Presentation materials prepared prior to meeting.

Example: The AP210 scope is very complex. This scope entails the details of both pysical and logical design of prited circuit boards. The boards contain many layers, some containing conductive material in logical patterns, some containing holes, some containing non-conductive materials. These concepts are difficult to picture for someone who is unfamiliar with the domain. The AP210 team prepared for the interpretation session by preparing samples of real PCB's in various stages of assembly and decomposition. This material made the interprestation session proceed much more smoothly due to its illustrative nature.

6.2 Phase 2: Interpretation

(This phase is also documented in "Guidelines for application interpreted model development". The steps differ slightly, and there is far more explanatory text for each step in the Guidelines document. The two need

to be coordinated and the text needs to be divided between the documents based on audience. It could be that the best solution is to have only one document.)

6.2.1 Concept mapping

6.2.1.1 Step 1: Assess ARM requirements concepts

Process: Assess ARM requirements concepts for semantic and structural correspondance to the ISO 10303 data architecture as discussed in 5.1. A quick way to accomplish with this step is to look at ~~as represented by~~ the integrated resource schemas, and query requirements experts to understand details not clear from the ARM or information requirement definitions. ~~Instance validation diagrams are a useful tool for understanding proposed mappings during this process.~~ This process consists of first a UoF planning model walk-through followed by a walk-through of the ARM. This is done without the use of the mapping table that is used as a tool for the detailed mapping that will follow.

Metric: Does the AP team understand and accept the chosen mapping?

Example: In AP 216, the Definition application object embodies a number of different entities in the integrated resources so it was positioned in such a way that the defined_for set was satisfied in the AIM. If it is a property_definition it is defined for a single characterized_definition but is a set in the ARM. The questioning determined that it was a set so representations could be shared and multiple lifecycle definitions could be specified for an item. Property definition provides the capability to aggregate multiple representations. ~~And you can have~~ Each instance of product_definition may have multiple instances of a property_definition ~~for a single product definition~~ to allow for different lifecycles, and the representation can still be shared ~~among all that~~. Shape is shared.
(Add details about the semantics of the ARM and add details about the semantics of the chosen AIM construct.)

6.2.1.2 Step 2: Identify the "centers of the universe" for the AP

Process: Identify the "centers of the universe" for the AP. What are the central ARM concepts?

Metric: Evaluate the responses to these questions:

— Does the AP team agree with the selected center of the universe?

— Did you consider other related APs?

Example: The center of the universe is the **Item** application object, which is subtyped into two concepts, **Ship** and **Moulded_form**. **Moulded_form** is the obvious center of the universe as it is the focus of this AP. However, in this suite of APs the center of the universe is always **Item**, as there are other subtype scentral to other APs in the suite in addition to **Ship**.

6.2.1.3 Step 3: Prioritize the UoFs

Process: Prioritize the UoFs. In the prioritization of the UoFs for determining interpretation sequence the following criteria indicate high priority. These are not hard and fast rules, simply guidelines.

— the UoF that is the center of the universe. This often sets a framework for the entire AP interpretation;

— the UoF that is most referenced by other UoFs, most depended on by other UoFs. It is best to know what the dependencies map to prior to mapping an application object or attribute;

— the UoFs containing a high degree of complexity. It is best to do difficult areas of the ARM while fresh and leave easy stuff and patterns for the AP team to do itself outside the workshop if time runs short;

— availability of domain resourcesfor clarification of ARM concepts. If key application experts are available only for portions of the workshop, their schedules have to be accommodated.

Metric: Does the chosen prioritization reflect the following:

— Does the AP team agree that this UoF represents the center of the universe?

— How many UoFs reference each other UoF?

— Do the AP team and interpreters agree that the selected set are the most controversial, complex?

— Does the schedule coincide with the availability of key resources?

6.2.2 Detailed mapping

6.2.2.1 ~~Step 4:~~ Identify AIM elements

Process: Perform detailed interpretation of the ARM. Walk through constructs from each UoF in a logical order, either progressing through diagrams or the mapping table entity by entity. The logical starting points are typically those used by the AP team to walk through the diagrams, while explaining the concepts therein. For example, the AP 216 team typically started by describing an entity that related to the prior model, or an entity that was central to the diagram, to which many other concepts on the diagram were related. For each application object, perform the following sub-steps:

Step 4.1: Verify the meaning of the attributes of the application objects by asking questions about the potential IR concepts till satisfied. There are a number of aspects to be considered in the questions chosen by the interpretation expert:

— What is the nature of the object from the applicationc standpoint (Do I understand the nature of the object within the context of the domain/application) ?

— What is its position or positions (there may be more than one) within the STEP architecture?

A typical concept that may have more than one position in the STEP architecture is one that has a role in both product and process data. There are instances in which a given thing (usually an application concept that maps to a product_definition) is used as a resource for some process. For example, in a manufacturing application for a particular enterprise, a tool may be both a product_definition (something that the enterprise designs), and a resource for a manufacturing process step in a process plan. If both of these views are important in the enterprise, then there would be a subtype of product_definition and action_resource developed for the AIM which captures the fact that the particular enterprise's product (the tool), in one life cycle stage (manufacturing operations, for example) is also an action_resource. Therefore, the concept would be positioned in both the product_definition and process_definition areas of the STEP architecture.

— What relationships does it participate in that would affect its semantics?

— How will the attributes of the object map into potential IR constructs?

Step 4.2a: Identify AIM elements for application objects and terminal attributes.

Step 4.2b: Identify key descriminati~~o~~ng elements in paths for assertions when both aplication elements in the assertion have been mapped. These descrimination elements are often strings that are defined to differentiate the role of a thing, or the type of a relationship. The elements may also be a required reference or subtype. The descriminating elements are captured in the mapping table as required elements of the reference path or mapping rules.

Step 4.3: See how mappings fit. If they do fit, proceed, if they do not, repeat step 4.2. Sometimes a mapping will seem to work on the surface, but in some of the more complicated mappings, example instantiations should be used to see if the mapping is really correct. Several iterations may be necessary before all of the factors such as participating relationships have been considered and the correct fit is defined.

Step 4.4: Stop when attributes and paths have been completed~~all matched~~.

Step 4.5: Document the mapping.

Metric: ~~*[develop metric]*~~Can the following be accomplished:

— Establish concensus among the team that the mapping is obvious.

— Example populations with data from the ARM validation report can be used to populate the AIM mapping.

Example: The following example from AP 216 executes the above steps interatively. *(Clean up example.)*

[a piece of the ARM diagram covering the ship_curve here would be very helpful]

Step 4.1: In 216 **Geometric_item** is a supertype of **Point**, **Curve**, **Surface**

Ship_curve.curve_shape -> **Curve**
Ship_curve is a type of **Curve** that is commonly used in naval architecture and that has an associated name.
curve_shape: The **curve_shape** specified the underlying geometric definition of the **Ship_curve**.
curve_class: The **curve_class** specifies the naval architecture category for the **Ship_curve**. The categorisation uses curves that are commonly required for the design definition of the hull moulded form.

Step 4.2a: Is **Ship_curve** a subtype curve from part 42?

Step 4.3: What does it mean to have a ship curve that points to a curve?
Does curve have shape?
Does curve have other attributes in addition to shape?
No, ship curve does not have shape, it associates a class name to one or many curve to shape.

Step 4.3: if not adding add'l info to shape then not a subtype of curve.

Step 4.2a: Is a Curve's existence dependent on its classification (has required curve_class) The answer is no, some curves may be classified

Step 4.3: A Ship_curve's existence dependent on its classification as evidenced by the structure. (has required curve class)

Step 4.2a: it has some kind of shape to it so it could be a shape aspect.

Step 4.3: To determine whether this is a shape_aspect:
Do you have the shape regardless of whether it is classified or not - yes
Would you have the same thing and not be concerned with the geometry - no -- always have geometry
Don't have concept of shape of something without its geometry so it is not a shape aspect.
Requirement not about specifying aspect of shape of ship regardless of representation.

Step 4.2a: Classification of a particular curve as a thing called waterline.

Step 4.2a: It is a classification concept, so mapped straightforwardly to group construct.

Step 4.4: Three subtypes to constrain name and items.

6.2.2.2 ~~Step 5: Document~~Record mapping in the mapping table

Process: Identify mapping rules that are needed. Map entities and attributes then leave assertions to the team as they are usually simply the path between the two. Document the rationale for key decisions.

Metric: Is there suffient detail documented for the completion of the details of the mapping table?

Example:

6.2.2.3 ~~Step 6: Document~~Record short form notes

Process: Document new subtypes and select types.

— Identify local rules for new entities as you go through the process.

Rules should be defined in natural language and coded in EXPRESS after the workshop.

— Note list of possible global rules for consideration at the completion of the interpretation.

Rules should be defined in natural language and coded in EXPRESS after the workshop.

— Document informal propositions for construct where it is not possible to write a formal rule. (For new or imported entities)

— Note imported entity modifications that need to be made in clause 5, such as clarifications and domain examples.

— Maintain issue and action list for ARM revisions

— Maintain issue and action list for IR/AIC SEDS or ballot comments.

Metric: Have the note takers been keeping their attention focused on the task at hand? Typical indications are that they will stop the discussions to ask questions, request for time to complete their documentation of the current mapping, or request clarifications.

Example:

6.2.2.4 ~~Step 7: Document~~Record clarifications of resource usage

Process: List of clarifications of IR and AIC construct usage to serve as aid for subsequent interpretation sessions and training of interpretation resources.

Metric:

Example:

6.2.2.5 Record ARM issues and requirements for clarification

Process: Identify ARM constructs and modelling formations that are unable to be explained well enough to complete a mapping. These constructs are either ill formed creating a problem for the ARM model expert to explain, or they are causing a semanitic mismatch between the requirements and the Integrated Resources causing the mapping to be overly complex. In this case something will be done with the ARM data modeling constructs to make a smoother mapping, or the ARM data modeling constructs may be simplified or removed altogether if they are deemed to be ambiguous or redundant.

Another reason for ARM changes is AP integration. Similar requriements may be specified in another ARM that have potential for overlap providing the capablity for the sharing of interpretations. In this case, one of the ARMs may be modified to reflect an agreement between the teams that make the requirements the same.

Metric: The Interpretation experts are unable to define a mapping from one or more ARM contsructs. All parties are able to identify specific ARM requirements that are causing the descrepancy.

Example: Anne should provide an example here from AP214. There were a few good ones.

6.2.2.6 ~~Step 8:~~ Document interpretation report

Process: Sometimes mechanisms are defined to satisfy certain requirements Those need to be captured. For example, representation.name being used as sequence number for the tables so that needs to be documented somewhere in the AP fundamental concepts and assumptions.

[Mark Palmer will have some material for this section and he can use AP 231 examples]

Metric:

Example:

6.2.2.7 ~~Step 9:~~ Document recommendations for new resources

Process: Recommendations for new AICs or IR extensions.

Metric:

Example:

6.2.2.8 Step 10: Document recommended population of resource attributes

Process: Recommend population of “for free” attributes in the Implementation and Usage Guide annex of the AP.

Metric:

Example:

6.2.2.9 ~~Step 11:~~ Provide review

Process: Provide periodic reviews of the AP team’s documentation of interpretation during the course of the workshop to ensure that the AP team is understanding the mappings and has sufficient information todocument the interpretation of the AP.

Metric:

Example:

6.3 Phase 3: ~~Documentation~~Consistency Review

The objective of this phase is to document the interpretation results by producing the formal document elements of the AP. No workshops are necessary for this phase of the AIM development process. The AP development team is responsible for the documentation of the AIM components. This phase is largely documented in Guidelines for application interpreted model development. This clause covers ~~only~~ the consistency checking that is performed at the completion of the interpretation.

The consistency of the interpretation is evaluated by means of several different consistency checks. Because the interpretation was performed UoF by UoF, and possibly over the course of several workshops, a consistency review must be performed after the interpretation is completed. The required consistency checks are documented in the following steps.

6.3.1 Step 1: ARM/AIM cardinality consistency check

Process: This check involves examining all the cardinalities in the ARM and ensuring they match the cardinality in the AIM. If any ARM cardinalities are other than zero, one or many (0,1,M), examine how the cardinality is reflected in the AIM. If the cardinality in the AIM is different, a global rule must be written to constrain the AIM cardinality.

— Make a list of all cardinalities in ARM that are not the typical 0,1,M (for example, L[1:M], attribute exactly 1, shipyard_designation_shipyard_name to organization is exactly 1.)

— Go to AIM and see how the above cardinalities are reflected in the AIM. If the AIM is less restrictive, you have to write a global rule to constrain it. If the integrated resources are more restrictive than the cardinality in the ARM, the only option would be remapping to a different integrated resource construct.

— Write global rule if needed.

— Add the global rule name to the list at the end of the mapping table.

— Note the sequence number of that global rule in the mapping table for all mappings it pertains to.

Metric:

Example: In AP 216, a product_definition that is a ship (rep_context.context_identifier = ‘ship’) has to have exactly one global_unit_assigned_context.

```
RULE single_global_units_context_for_ship FOR (product_definition,
representation_context);
WHERE
   WR1: SIZEOF (QUERY(ship <* QUERY(pd <* product_definition |
        SIZEOF(QUERY(ctxt <*
```

```
        product_definition.formation.of_product.frame_of_reference |
        ctxt.name = ‘ship’))>=1|
        NOT( SIZEOF(QUERY(pd<*
        USEDIN(ship,‘SCHEMA.PROPERTY_DEFINITON.DEFINITION’)|
        NOT(SIZEOF(QUERY(pdr <*
        USEDIN(pd,’SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION’))
        pdr.used_representation.context_of_items.context_identifier=’ship’)
        AND (‘SCHEMA.GLOBAL_UNITS_ASSIGNED_CONTEXT’ IN
        TYPEOF (pdr.used_representation.context_of_items))))=1)))=0)))=0;
END_RULE;
```

In this rule:

— get all products in context of ‘ship’ to use as basis;

— for each ship check that every property_definition for each property_definition_representation used has exactly one context_identifier;

— for each property_definition_representation, check that there is exactly one global_unit_-assigned_context with name of ‘ship’.

(If something maps to an ~~or~~“OR” of all the subtypes like definition.defined_for then one has to check all subtypes.)

6.3.2 Step 2: Mapping table consistency check

Process: In this check, the mapping is examined to ensure that different ARM constructs are not mapped to the same instance of an AIM construct. First, identify all ARM elements that map to the same AIM element and determine whether they map to the same instance. Examine the reference paths, particularly the path constraints as well. If multiple ARM constructs map to the same attribute of the same instance, the mapping must be changed. Raise issues against the mapping ~~to the interpreters~~ and ~~re-map~~ reconsider the mapping of the problem areas.

Example: In AP 216, hydrostatic_element and moulded_form both map to shape_aspect.description. Because these will be different instances of shape_aspect, there is no problem.

Metric: All of the rows of the mapping table have been checked, issues raised if necessary, and resulting modifications identified and incorporated into the mapping table and short form.

Example: In AP 216 approval_event and approval_history both map to approval_status.name. Is the intent to have these map to the same instance? No, they are two separate instances as they have a relationship entity between them.

Make sure two things aren’t unintentionally the same, or that the same instance is being constrained to have different values in different mappings.

6.3.3 Step 3: Supertype and subtype consistency check

Process: This check examines the relationships ~~between~~among subtypes created in the AIM and their relationships to other subtypes of the same supertype that would be imported into the AIM EXPRESS expanded listing. Examine whether the supertype can be instantiated itself. Write global rules to restrict instantiation of subtypes. The decision in this check is twofold:

— Should there be a subtype mandatory rule written in which a supertype entity may not be populated within the scope of the AP without being populated in one of its subtypes?

— Should there be a subtype exclusive rule written in which a supertype entity may be populated indepently, but constraints are necesarry to limit the combinations of the subtype entities that may be populated?

The integrated resource structure has default ANDOR relationship between subtypes. If that is the intent, no rule is necessary.

Metric: Each entity that has a subtype declared or imported in the AIM short form shemas has been evaluated, the necessary rules have been written in the short form, and the rules have been indicated in the RULES column of the mapping table.

Example: During the interpretation process for AP 216, several subtypes of Group were created: Floating_position, Class_notation, Ship_point_class, Ship_curve_class, and Ship_surface_class.

Are there any allowable complex instances? Can something be a Floating_position and a Class_notation at the same time? Floating_position and Ship_point_class? No allowable complex instances, they all have to be instantiated one at a time.

Is there a requirement to instantiate Group itself? No.

All instances of Group are to be one of the subtypes. Only.

Write a global rule to constrain.

```
RULE subtype_mandatory_group FOR (group);
WHERE
 WR1: SIZEOF (QUERY (grp <* group |
    NOT(SIZEOF(['SCHEMA.FLOATING_POSITION',SCHEMA.CLASS_NOTATION',
    'SCHEMA.SHIP_POINT_CLASS','SCHEMA.SHIP_CURVE_CLASS',
    'SCHEMA_SURFACE_CLASS'] *
   TYPEOF(grp)) =1)))=0;
END_RULE;
```

If you want to instantiate the supertype also, the rule would be different. It would allow less than or equal to one of the listed subtypes. When 0, group would be instantiated.

```
RULE subtype_mandatory_group FOR (group);
WHERE
 WR1: SIZEOF (QUERY (grp <* group |
    NOT(SIZEOF(['SCHEMA.FLOATING_POSITION',SCHEMA.CLASS_NOTATION',
    'SCHEMA.SHIP_POINT_CLASS','SCHEMA.SHIP_CURVE_CLASS',
    'SCHEMA_SURFACE_CLASS'] *
   TYPEOF(grp)) <=1)))=0;
END_RULE;
```

6.3.4 Step4: Value domain consistency check

Process: This check results in rules constraining a string population to be of a defined set. Examine the mapping rules that were identified during the interpretation and determine whether you know the complete set of populations of a string attribute. If so, you can write a global rule that requires the string value to be limited to that set. Such a rule cannot be written for open-ended enumerations. The constraint could also be written as a local rule if the restrictions all apply to one mapping and a subtype was created as a place holder for other local rules.

Enumerated value definitions: must also repeat or reference definitions for the enumerated values from clause 4 in clause 5 where the global rule is documented.

Metric:

Example: In AP 216, approval_status.name may be either approved, unapproved, noted, rejected. This same set of enumeration values appears in both approval_event and approval_history.

Is this a closed set of strings for this AP? Yes.

Write a global rule to restrict set of strings.

```
RULE restrict_approval_status FOR (approval_status);
WHERE
 WR1: SIZEOF(QUERY(as <* approval_status |
    as.name IN ['approved','unapproved','rejected','noted'])))=0;
END_RULE;
```

6.3.5 Step 5: Mapping rule consistency check

Process: This check is to ensure that string values in the mapping rules found in reference paths in the mapping table are consistent with the types and values in the ARM and AIM. If inconsistencies are found, the string values in the mapping rules shall be corrected.

Each reference path that contains a mapping rule constraining a string value is compared with the requirement in the ARM to ensure consistency of intent and, possibly, terminology. If the string value constrained is in an entity from the integrated resources for which a subtype has been declared, and that subtype is called out in the reference path, the string value is checked for consistency with any rule in that subtype entity. A typical type of constraint that would need to be checked is a relationship type string that is populated in the name attribute of a relationship entity.

Metric: Each reference path in the mapping table has been checked against both the ARM and the AIM.

Example: In AP 216, approval_relationship maps to approval_relationship with .name = 'context constrain approval'. How was this string derived? It is the role of the relationship--being used to constrain the context for the approval. Related is constrained within the context of the relating. Readjust name to "context constrained approval"

6.3.6 Step 6: Context consistency check

Process: In this check, guidance for population of values for application_context_schema attributes is derived from usage scenarios. Examine the activity model. If the same data set can flow in two or more paths, then multiple values may be needed. If a data set is only valid in one environment, the values may need to be constrained on a per conformance class basis.

Metric: Each conformance class has been examined against the usage scenarios. Guidance as to the population of these attributes can be found in either the mapping table or clause 5.2.2 Imported entity definition modifications.

Example: *An example was documented in clause 5.10 of this document; should it be moved here?*

6.3.7 Step 7: Dependent instantiability cardinality consistency check

Process: This check ensures that needed constraints on the instantiation of entities are written. This check is performed once the AIM is complete.

Examine USE FROM statements for entities that are not allowed to be independently instantiated. Write rules constraining the instantiation of these entities. Entities that are subtypes of representation_item or its subtypes do not require dependent instantiation rules as such a constraint is locally defined in representation_item and inherited by all subtypes. As they cannot be independently instantiated anyway, provide only the minimal set of use statements, i.e., explicitly interface only the bottom leaf subtypes.

> NOTE - Some entities are independent by definition. These must be explicitly interfaced to bring them into the AIM. They are not referenced by other entities and it is, therefore, incorrect to write a dependent instantiation rule for such entities.

Add USE FROM statements for any entities that are intended to be independent but that were not already explicitly interfaced in the execution of a previous step. Some constructs may be implicitly interfaced that are required to be independently instantiable. These entities must be explicitly interfaced. Independent entities are those entities that may be instantiated without serving in the role of another entity's attribute.

Metric: Each entity defined in a USE FROM has been considered, dependent instantiability rules have been written, and the constriants have been indicated in the RULES column of the mapping table.

Example:

7 Interpretation process details Anne/Dave

(This clause is very rough. I believe the intent is to provide a textual description of the rationale and motivation for the various aspects of the interpretation process: to include heuristics rather than simply steps; to provide interpreters with a framework for making their decisions. It may well belong as clause 6 with the detailed steps, metrics and examples following in clause 7.)

7.1 Analysis of ARM

There is only one condition where the interpretation process would be mechanistic--if the process for developing ARMs would be as formal as the process for developing the AIM. However, if all ARMs were defined at that level of formalism, then there would be no need for AIMs, no integration across application protocol boundaries, and correspondingly, no hope for sharing of product data among implementations of different application protocols.

The differences between ARMs are not purely terminology, they also reflect a different viewpoint. An industry that calls a product a part will have particular data that they hold about that part and particular relationships that differ from other industries.

Knowledge--not reasoning. Mitch says interpretation is instinctive, he cannot state reasoning process.

The integrated resources are an articifially constructed vocabulary for talking about products. The interpretation process looks at set of requirements with the objective of discovering the fact that certain concept in those has same concepts as in the integrated resources.

This is what the IRs mean at the level of product data (documented in the IRs). This is what those resources mean in particular usage constructs (documented in APs). In some places we have the same meaning and same name in different contexts (documented in AICs).

Mapping of supertypes different from the mapping of the subtypes, which is not nice, but caused by the following problem: 216 has abstracted things that are very different things in nature, abstraction strategy is a different strategy in 216 than in IR structure; the subtypes carry the real meaning and are very different things, so 'is a' relationship of ARM can not be taken over into AIM

7.2 Selection of resource constructs

It is possible to have the mapping of a supertype be different from the mapping of the subtypesdue to the following: 216 has abstracted things that are very different things in nature, abstraction strategy is a different strategy in 216 than in IR structure; the subtypes carry the real meaning and are very different things, so 'is a' relationship of ARM can not be taken over into AIM

When an object has lots of attributes, look at the attributes first to see what they map to. The object itself will map to an and of the things the attributes map to?

Item is abstract. Item was mapped to a select type. Typically would have mapped the abstract supertype to an OR of what the subtypes were mapped to. In this case picked something more generic.
What happens if we find another subtype of item that does not turn out to be under the select list of characterised_definition. -- would have to change mapping. However, all the subtypes of item have properties and that is consistent to what characterised definition.

In general, when you map an abstract supertype in an ARM, you map to an OR of the AIM elements of the subtypes. If there are multiple levels of abstract stuff, map down to where it gets non-abstract? (In 216's case, however, since we are trying to be generic, we will probably map to a more abstract construct then constrain it with mapping rules.)

Rules:

— Map to supertype and all of subtypes map to supertype;

— Map to supertype only; map to supertype in vertical bars;

— If you have an abstract supertype you have to map to the OR of the subtypes;

— To be generic, we mapped to supertype with mapping rule;

— Global rule check at end needs to see if we need subtype mandatory rules.

In general, the ARM supertype and subtype are mapped to the same AIM element or the subtype can be mapped to a subtype. There is a justification for not doing that when there are different attributes on the subtypes. Seemed to be a bad thing to map to different things This is because there is no assertion between the subtype and supertype in the ARM so if they map to different things, then we have a mess.

Because it is abstract it can be mapped to a non-instantiable select type.

Product_definition through path with a_c_e.name of ship.

When there is a subtype/supertype tree (ship has subtypes of 5 types of ships, categorization structure). This would be accomplished with product_category. Product category may not even come into the scope of this AP.

7.2.1 IRs

The criteria for selecting integrated resources during interpretation are based on the design architecture, the definitions, and the intended use of the IRs. The IR design architecture seperates the product data concepts and product data management concepts. Within product data, concepts are seperated into 'context', 'identification', 'definition and property', 'representation of properties', and 'presentation of the property representations'.

The first criteria is to determine which one of these concepts a requirement belongs to. For example, a name and value pair indicate that it is a concept of 'representation'.

Once this first criteria is met, the second criteria is to examine within this general concept, how do all the attributes match, so that the specific entity or attributes can be identified. The third criteria is to make sure that all mandatory attributes on a given ARM entity are in fact also mandatory from the matching IR constructs. If this is not the case, EXPRESS subtypes are created to specify these rules.

In majority of the cases, there is no room for preference or individual's judgement. The IR architectural concepts distinguish product data in a much finer manner than what we are used to (people tend to lump a lot of concepts together). So, if a selection is properly made of the IR architectural concept, the details usually fall very neatly in place.

Once in a while, there maybe judgements required and usually it is because the requirements are not clearly defined. Because of these occassional judgements, it is always a good idea to have more than a single interpreter available to reach agreement on the judgement.

An important structural factor when selecting an integrated resource is whether the construct can support the requirements for collections of related items being associated with the same item and at what level instances can be shared. For example, property_definition is one-to-one with product_definition but for different life cycle stages the same representation of the property can be shared.

Clarification of IRs

1) The issue is that the integrated resources appear to be documented as entities which in themselves have useful semantics upon which implementations could be based. However, the text definitions of the constructs are generic so that different concepts from different ARMs may be mapped into the same IR construct. The text definition of the IRs would better serve AIM development if the text definitions of the constructs were based on the usage of the construct in the interpretation process. These definitions could be improved by describing the important structural aspects, i.e. the relationships and cardinality of those relationships, that the construct is designed to support in addition to the concept and characteristics of the entity or type. For example, the 216 concept of item which might appear to be a product was in fact mapped to characterized_definition because it is a select type of things that can be definitions.

2) The difference in the usage and concept of product_context and product_category needs clarification. This usage also needs to be consistent among APs even in the case where the APs have only one product “type” (this is an AP Interoperability and data sharing environment issue as processors and databases of multiple APs need some mechanism for filtering out products and at lower levels product_definitions and properties that the processor understands vs. those that are not common between the two APs).

3) For assignment concepts from the management resources there is an issue that the strings used for the name of the entities do not carry the usage of the assignment and that the same string is used for entity names and attribute names when different concepts are intended. An example is name_assignment and the proposed identification_assignment in 41. The definition for name_assignment is “A name_assignment is an association of a name with product data”. This definition adds nothing to the understanding of the entity or its usage that is not evident from the name of the entity. The definition also does not explain what a name is, why a name would be associated to product data, why some product data already has a name attribute and how that

attribute is different from a name_assignment or the difference in the usage of name and id. Clearly these could not be added as normative text, assuming the IRs remain IS documents, but notes, examples or technical discussions in an annex could provide this clarification.

In the 216 case they have abstracted concepts that are very different in nature which is a different abstraction strategy than that in the IRs. This causes the selection of IR constructs which do not have subtype/supertype relationships to support ARM concepts that do have the subtype/supertype relationship.

Guidance needs to be given to implementors of the APs describing the concept that the mapping table is a key aspect of the AIM describing how the selected IR constructs are to be populated.

Some guideance for intepretation needs to be added for the cases where ANDORs are used. Reviewing the IR constructs would not be sufficient to understand what useful complex entities appear in AIMs vs. those that are possible as the supertype constraints are not specified in the IRs so that they may be as generic as possible.

7.2.2 AICs

(Process for selecting an application interpreted construct, questions to ask of the model.)

What to do if you want to constrain the usage of the AIC within the AP--can write global rules in the AP short form that restrict usage.

7.3 Identification of new resource requirements

7.3.1 IRs

(Can't remember what was intended here.)

7.3.2 AICs

(Can't remember what was intended here.)

7.4 Subtyping Anne

(Purpose of creating a subtype within an AP.)

(Situations that warrant creation of a subtype--characteristics of requirements that are indicators; questions to ask of the model.)

7.4.1 Creation of subtypes

What are general rules for determining when it is appropriate to create subtypes?

Don't create subtypes just to give population guidance; only create subtypes if there are specific restrictions/semantics to capture a new concept corresponding to the ARM requirements

-> The MORE subtypes the LESS INTEROPERABILITY!

If there are mandatory attributes with closed sets of values then you have to write a subtype with the rules

Creation of subtype for management resource constructs. Annex E. in part 41. Identification_assignment is new management resource construct. Reference info in 41. Supports assignment of multiple ids to with different roles. When new constructs are added that have natural id, the entity will be defined with attributes.

Subtype when there are additional constraints that must be added as global rules. When there are simply population constraints use mapping rules. Concepts with their own semantics and constraints require subtypes.

7.4.2 Conventions for subtyping the management resources

It makes sense for management resource subtypes to check in general if there is a need to write local correspondence rules to restrict required/allowed combinations of role names, types of items andor types of assigned things

e.g. ship_building_group_assignment < = group_assignment

WR: coordination of items in grouped_item_select with the type of assigned group (points correspond to ship_point_class etc.)

Consistent naming of management resource subtypes and corresponding SELECT types: if semantics of the specializations is the same, there should be a consistent naming of the SELECT types, but it has to be checked carefully if the semantics is really the same.

— need to document naming conventions for management resources specializations and their corresponding SELECT type

— management resource entity specialization should be unique within the scope of an AP as they usually have different semantics; they should carry the rules e.g. to specify correspondence of role names and the types of items; in case the intended semantics is exactly the same, there might be a sharing of the concept; this has to be decided on a case by case basis

— need to document how and where to specify correspondencies between role names and types of assigned object and/or items in the SELECT (when specify as local/when as global rules): -> different types of local/global rules that might be applicable for specializations of management resource entities

— guidance needed when to create different subtypes of management resource entities and when to use the same (-> when to consider as different concept?? e.g. when different local rules are existing, e.g. involving moulded_form which might not be in the other ship APs) -> also consider maybe different solution for a suite of APs

7.5 Constraints Anne

(Purpose of creating a constraint within the AP. Purposes of the various types of constraints.)

(Situations that warrant creation of constraints--characteristics and questions of requirements.)

Explain the usage of local versus global rule: put constraint as constraint on the instances of an own subtype (local rule), if possible; use global rule in other cases e.g. to restrict instances of the IRs or specify constraints that affect a set of instances

The need for local and/or global rules is often caused by requirements that are rather implicit in the ARM or even totally missing, but are part of the explanations given by the team prior to the mapping -> need to document these identified restrictions in the ARM

e.g. team did not capture the need for the local rules on moulded_form entity: why local rule needed that moulded_form is always the shape_aspect of a ship? -> because semantics of the ARM was understood the way that in case there is a moulded_form, it always has to be related to a ship (application_context_element.name of the product_context = 'ship')

Need for additional constraints when mapping to AICs:

QUESTION: geometric AICs (e.g. geometrically bounded surface) allow for mapped_item as rep_item, which is not explicit as requirement in 216 ARM so far; should a contraint on the use of this AIC be considered in the scope of AP216?

->answer was that the requirement was not explicitly stated but it wouldn't make sense to restrict

-> Should the requirement be made explicit in the ARM

-> CLARIFY IN GENERAL what happens when an AIC has to be further restricted by the AP (use AIC at all with appropriate GLOBAL RULES??, or just copy needed portions into AIM??, restrict population via mapping rules???, ...)

DECISIONS FOR WRITING LOCAL VERSUS GLOBAL RULES:

- if the decision has been made to generate an AP specific subtype, constraints on the subtype instances or the usage of the subtypes should be formulated as local rules if possible;
- if constraint can be formulated in EXPRESS, but not as local rule (either as the constraint is on entities imported from the IRs or it involves multiple instances of the AP specific subtype) write global rule
- if constraint can not be formulated in EXPRESS, write informal proposition preferrably on AP specific subtype, or, if the constraint is valid for all usage cases of an IR entity withi the AP, in imported entity modifications
- for the same type of constraint write one global rule with separate WHERE clauses instead of multiple global rules with one WHERE clause

GLOBAL RULES: are often caused by cardinalities in the ARM (also INVERSE)

-> after mapping (usually done as homework by the AP team): go through the mapping and identify need for global rules corresponding to the cardinalites specified in the ARM

-> guidance needed regarding what types of rules are applicable

- check after the mapping, which mapping rules can be described as global rules in the AIM schema

AP SPECIFIC SUBTYPES WITH MORE GENERIC CHARACTER (e.g. table)

- e.g. mapping of offset_point_table_model and row_of_offset_point_table:

  216 subtype: table_representation_item <= [rep_item] [rep]

  WR1: all rep_items are of type table_row_rep_item

  WR2: all table_row_rep_items contain the same number of items (rows all have the same size)

216 subtype: table_row_representation_item < = [rep_item] [rep]

WR1: each table_row has to be associated to at least on table_rep_item

WR: all rep_items are of type point (WRONG, SHOULD NOT GO AS LOCAL RULE)

In this particular case the requirement to represent tables with its rows is not a new one; has occurred in several APs before which seems to indicate the need for a more generic concept that is sharable among several AP: possibilities are

-> to extend the IRs or

-> if a mapping can be found to define a more generic concept that could be included in an AIC -> in the case where a more generic concept is needed omit those local rules that only seem to be appropriate for the very specific case (e.g. see above) and formulate these as either local rules in the entities that use the more generic one if that's possibles, or if that's not possible specify the specific constraint as global rule

QUESTION: how to handle correspondency between the existence of an offset (longitudinal_position) and the allowed spacing table usages (as local, global rule or informal proposition?) -> done in specific entity spacing_grid_definition, which leads to more complex local rules, but allows the table_rep_item and table_row_rep_item definition to stay more generic

why not subtype the generic table_rep_item and table_row_rep_item for the specific need?? -> restrictions of tables and table_rep_item only in their use by the spacing_grid_definition, not seen as specialized concepts

DECISIONS FOR WRITING INFORMAL PROPOSITIONS:

if a restriction which is part of the definition of a new AP specific subtype cannot be formulated as local rule or if a restriction which is part of the modification of an imported entity cannot be formulated as global rule

e.g. order of the rep_items within a table row has to be maintained by the implementation -> either by the name attribute or by a rep_relationship with .name = 'row_sequence' with .relating as preceeding and .related as the succeeding representation

216 team discussion: sharing of rows might not be such an important requirement to justify the overhead of these rep_relationships: -> have row_id map to rep.name and sequence_id map to rep_item.name; if sharing is required, use rep_relationship with .name = 'same' between same rows occurring in different tables

CONSISTENCY between mapping table rule in mapping of longitudinal_position.offset and local rule -> Consistency check necessary after mapping if e.g names of the measure_rep_items consistent in mapping and local rules

MAINTAIN LIST OF POSSIBLE CONSTRAINTS

- e.g. mapping of moulded_form_characteristics: rep with .name = '....'

During the course of the interpretation of AP216 several potential global rules were mentioned. The plan is to leave development of global rules till after the interpretation in complete. The AP team needs to take notes on this! and be sure to get back to them at the end of the interpretation.

Usage of local vs global vs mapping rules. Need to have at least one representation, where we have our own subtype, first try to write it in the subtype, If can't write in subtype write in global rule.

When it is impossible to write a rule in the express or in the mapping table, it is written as an informal proposition.

When to use one global rule with 2 where rules or two global rules? One constraint on two entities best documented as one globla rule. If there are unrelated constraints they go in separate global rules.

Is there any guidance to derive on inverses? Existance of inverses is often an indication that a constraint is needed. Unique maybe less so.

When entities mapped that are intended to be in the AICs, they are mapped generically and they try to use maping rules instead of local rules in the subtype. The generic capability to support tables could be an AIC or could be an extension to the Irs.

need for AP team to check consistency between local rules and mapping rules

>are there cases where the same constraint is to be written as both a local rule and a mapping rule?

usually when it is a representation, they resist creation of subtypes, particularly when it is a collection of measures.

General resistance to subtyping in general, representation in particular, because the collection of things called a representation is very arbitrary and often not complete. With a subtype, you have restricted what it is called.

When all 5 ship APs are looked at, if there is agreement that something is always called a representation, maybe a subtype will be written, or at least a global rule to restrict the number of rep_items in it. If ship application experts can agree that a ship rep contains always and only these sets of things, then a subtype could properly be created.

7.6 Mapping table rules

(Purpose of creating a constraint within the AP. Purposes of the various types of constraints.)
(Mapping table is the source for population rules.)
(Situations that warrant creation of constraints--characteristics and questions of requirements.)

USAGE of MAPPING RULES
Guidance for population of the AIM model according to the ARM semantics; often need for mapping rules that only restrict the population to capture a particular application object semantics, but is not valid regarding all possible AIM populations

Don't 'waste attributes' that could be useful for an implementation if not necessary; e.g. for mapping of moulded_form_relationship the type of relationship can be determined by putting mapping rules to restrict that the type of the relating and the related is moulded_form -> no need for a mapping table rule to restrict relationship.name to be 'moulded_form_relationship'

MAPPING RULES VERSUS CREATION OF SUBTYPES
QUESTION: where define that rep is representing the principal_characteristics and that this rep contains … rep_items of type … with a name of … (design_draught, etc.)
-> why not define an own subtype???

-> interpretation does not like to create subtypes especially for representations that contain measure_rep_items; contained rep_items might be arbitrarily chosen and not complete, revisit after all ship AP have been mapped and if there is uniqe understanding about meaning and amount of rep_items creation of subtype possible

-> at least mapping rule necessary to restrict rep.name to be 'e.g. principal_characteristic'; global rule possible to restrict the amount, type and name of parameters

MAPPING RULES or other CONSTRAINTS that don't show up in the ARM:

Often requirements are not explicitly formulated, that show up in the mapping table or even just in the AIM (e.g. requirements for specific measure units). These requirements should be documented in the ARM and show up in the mapping tables, often as mapping table rules. Selection of units e.g. through mapping rules if there is no explicit application object for all the different units required.

MAPPING RULES that only show up in the mapping tables but nowhere in the AIM:

Experience is that implementors often just look at the EXPRESS AIM and not at the mapping table/ARM . Therefore there are problems in correctly understand how to correctly populate the AIM according to the ARM semantics and the population guidance given just as mapping table rules (mapping rules that don't show up in the AIM).

It might help to have documentation available in the AIM (or maybe recommended practices) that explain the meaning of e.g. string values (locally on new subtypes or in imported entity modification sections) and relate them back to the requirements.

** don't use underscores within strings in mapping rules

FREQUENTLY REQUIRED MAPPING TABLE RULES:

guidance necessary for mapping rules to restrict or allow different rep_contexts: e.g. different subtypes of rep_context are often forgotten to be included in the AIM, e.g. global_unit_assigned_context, .. -> team needs to check during or after the mapping that the right types of rep_context are included in the AIM and properly reflected in the mapping tables

mapping rules often required in the area of measures and units (e.g. for a rep_item representing a length); if AP specific subtypes existing then also put in local rules

-> for AIM inclusion of appropriate measures and units clear statement of requirements is useful in order to be able to document the usage of all required units in mapping table

general guidance required for usage of type_qualifiers versus rep.name/rep_item.name and for when it is necessary to put the mapping rules for the type_qualifier.name

identify the context of the AP: context of the AP, context of the product_definitions (life cycle stage)

-> during mapping especially when using product and product_definition, it should always be reflected what the context of these things are in the scope of the AP, if the AP is embedded in a suite of APs, also in the scope of the suite. product_context usually carries the context for the whole AP, product_definition_context the lifecycle oriented context for the particular definition.

-> show the allowed/possible contexts in mapping rules at the appropriate place (in 216: ship -> appl_context_el.name = 'ship', mapping of design_definition should restrict product_definition_context.lifecycle_stage to 'design')
-> at the end of the process, check, if these constraints on the context attributes (written as mapping table rules) can be written as local or global rules

***Population guidance is found in mapping rules in the mapping table reference paths. This note needs to be added to the methods documents in general. It is also valuable guidance for the implementors.

7.7 Documentation of interpretation meeting results

(This section to state how to document

— the info that will be needed to complete the AP documentation outside of the workshop;

— the interpretation for AP experts to facilitate their inderstanding of the AP and ap Implementors, and document tracceability of requirements between the ARM and the AIM.)

Interpretation is documented in several places--in the AP as a mapping table, AIM short form, new entity definitions and imported entity modifications. The rationale for the interpretation is documented in the interpretation report section of an AP's Validation Report. The following describes what should be found in a typical Interpretation Report. Some of this content is dictated by the "*AP qualification manual*," "*Guidelines for development and approval of STEP application protocols, version 1.2*", and the "*Guidelines for AIM development.*" (If other items are added due to the workshop, state that here.)

The following are specifics to include in the Interpretation Report of the AP Validation report. Ideally, these should be documented at the interpretation workshop by the team doing the interpretation of the AP. The AP development team and the interpretation team shall agree that all semantics of the information requirements specified in clause 4 of the AP have been satisfied by clause 5 of the AP. The Interpretation Report must capture the rationale on all specific that are outlined 4.4.1 and 4.4.2 on the "*Guidelines for development and approval of STEP application protocols, version 1.2*". and in clause 2.2 and clause 3 of the "*Guidelines for AIM development.*" It should be noted that severals issues current exist with the contents of clause 4.4.1 E as written in the AP guidelines document version 1.2. The issue purposed solution states that the interpretation team is responsible for documenting the interpretation report and providing it to the AP development team for inclusion to the AP Validation report.

1. When multiple paths are possible for a mapping of an application element (AE) a complete and unambiguous description of why a specific path was chosen over any other possible path. This description should be in layman terms so that the descriptions are understood by domain experts.

2. When subtyping of an IR entity is required to support the AE requirement, describe the rationale for the subtyping and describe what changes shall be to the entity, i.e., attributes and rules.

3. When an AE can not be mapped to an IR entity a complete and unambiguous description of the problem along with a proposed solution shall be written in the Interpretation Report and submitted to SEDs. The report should identify a point of contact, the projects/WGs/ that will correct the problem, and state the

estimated completion date of IRs in question. When this condition exists one or more issues shall be recorded in the ISSUE LOG for the AP. A NO MAPPING statement in the mapping table is not a possible solution.

4. When a specific path is chosen for interoperability with other APs, describe how interoperability will be achieved or enhanced by this specific path. If specific rules are required to insure that interoperability will occur they should be stated in the Interpretation Report of the AP.

5. The Interpretation Report shall state specifically the pruning that has occurred, i.e., SELECTs and SUBTYPE/SUPERTYPE trees, and state why that pruning was necessary. If the requirements are not explicit in clause 4 the report should identify the deficiency and state how to harmonize clause 4 with the resulting interpretation.

6. When a mapping table rule is required a description of the rule must given and why it is necessary.

7. If the most of the information requirements stated in clause 4 of the AP are generic in their descriptions and definitions the Interpretation Report shall state what determines this to be an application protocol and not and IR. Generic is defined as groups of entities and/or UoFs that will serve the needs of many other APs in other disciplines/industries.

8. Neither of the Guideline identified above provide guidance on ENUMERATION types. If it is possible to prune items from an ENUMERATION, the rationale for such pruning shall be stated in the Interpretation Report of the AP.

Charleston Notes (what to document during):

1) AP TEAM should capture questions about quality of definitions in resources when team is not clear on def (ex: symmetric_shape_aspect) and discuss with IR owner. AP Team should feed question to SEDS if is IS and as Ballot comment if at FDIS or before.

In this case definition should be enhanced to say can be half that can be mirrored to become a symmetrical object, as well as an entire object that has property of being symmetrical.

2) New Measures_with_unit.value_component and .unit_components, new Roles on Assignment Subtypes, and new applicationContextNames, determined in an Interpretation Workshop should be fed to ‘standard list’ for APinteroperability. Also should include new types of _assignment entities and _item SELECTs created in the APs.

3) Keep list of possible Global Rules as do interpretation so can evaluate at end of ARM to see which Rules need to be written.

Notes:
EXPRESS-G diagrams help! People in the workshops do not know the integrated resources as well as the interpreter. The diagrams allow people to focus on the discussions. Greg Paul suggested: Parallel explicit pictures of the ARM and the AIM with double headed arrows representing the mappings. *(Greg to send to Allison)*

People assigned to capture stuff should be famililar with how to document it so the documentation can be written in as close to final form as possible.

Note changes to the ARM. and reasons why.

Need to get closure on questions related to the integrated resources.

Broad issue is there are no custodians for the IRs that are technically competent to answer these questions.

AP project is responsible for:

— obtaining clarifications from part owners;

— filing formal ballot comments or SEDS issues;

— assuring issue is resolved correctly or changing interpretation.

8 AIM quality metrics

This section deals with assessing the quality of the AIM with respect to how well it represents the requirements that are defined in the ARM. When application experts develop and review the ARM with industry experts within the domain defined by the ARM, many underlying concepts are assumed in the development of the ARM. Also many concepts that are not explicitly represented in ARM are assumed within a discussion (and a data exchange). An example of an assumed concept that may not be explicitly represented in an ARM that is an assumed concept may be units associated with a value for a length. ISO 10303 requires explicit definition of the length so that data exchanged with ISO 10303 has explicit information regarding the units of numeric values (e.g., inches, feet, centimeters, etc.).

As part of the AP Development process, the domain experts were to have utilized instance population to verify that the ARM represented. This instance population should be brought to the Interpretation Workshop to help interpretation personnel in the development of the mapping table.

Validation of ARM with instance population. This is useful tool to bring to the interpretation workshop to check the proposed mappings.

8.1 Traceability

During the interpretation process the interpretation personnel are given detailed information relative to the ARM. Interpretation personnel will probably ask a lot of questions to clarify the concepts in the ARM to ensure that the interpretation personnel have sufficient understanding to map the concepts into the ISO 10303 data architecture. If during this process additional detail is brought out that should be documented in the ARM, this information should be incorporated into the ARM as ARM clarifications. If the AP development team perceives that this information is so basic that it not required in the ARM, then it should be documented in the AIM Fundamental Concepts and Assumptions or the Imported Entity Definitions Clarifications or the

New Entity Definitions in the AIM or in the Interpretation Report. It is recommended that the information is documented in a place where an AP implementor could access the information. ISO 10303 requires a clear definition of all concepts and information (e.g., units) and requires this for unambigous transfer of data.

After the mapping is complete, there will be many attributes in the ISO 10303 integrated resources that do not have a value mapped to them or a mapping rule associated to them (relative to what the value should be for consistency). These "unfilled" attributes are not required for exchange of information and the AP Team should recommend what information is placed in these "unfilled" attributes. The reason that these attributes are "unfilled" is that the ISO 10303 integrated resource data architecture has identified the need for the "unfilled" attrributes in many data exchanges. In some cases, entire ISO 10303 integrated resource entities within a path do not have a value. In concept, these ISO 10303 integrated Reesource entities collapse into a single concept within the AP domain. The "unfilled" attributes do not fulfill a requirement within the AP and therefore should be primarily ignored within a data exchange.

The AP Team should put together a set of information that may be placed into the "unfilled" attribute values and place this information in the AP Usage Guide (Appendix XXX) or as a separate report that has these recommendations within it. The following is a guide to the "completion" of these attributes.

1) When a xx.identification is not specifically populated in the mapping table, it should be populated by the processor with a value that will give the entity some type of identification. For example, the first product_definition.identification in a physical file could be "PD1", the second product_definition.identification could be "PD2", etc. The value should not be blank or null or the same value for all instances.

2) When an xx.description is not specifically populated in the mapping table, the recommended practices (or another document) should identify some recommended values and may leave blank or null.

3) When other attributes are not specified such as xx.market segment, the AP Team may be able to logically recommend populations to the information.

The information that is contained in the "unfilled" attribute information in the precceding paragraphs do not contain information that is traceable from the AIM back to the ARM. This information should be so noted where the recommendation of the "unfilled" values are documented in the AP or the Recommended Practices Report.

One example of traceability is in the area of unit definitions. If the ARM has only a requirement for date unit information, then the AIM should not include information or entities that contain length measure or volume measures, etc.

Another example of traceability is in the are of enumerations. If the ARM has a defined set of enumerations that map to an AIM element that is used in many different contexts, then the possible mappings for the enumeration are contained in the mapping table. In the import entity definitions sections of the AIM, it is advantageous to document what some of the enumerations are and where the enumerations came from in the ARM for traceability and as a guide to the implementors of the AIM.

In summary, there should be traceability from the ARM to the AIM. There are two exceptions to this case:

1) The ISO 10303 integrated resources have a requirement to identify the application protocol that the data is generated from (AIM element called: application_protocol_definition),

2) Where there is additional richness in the ISO 10303 integrated resources that is not required for the AP (i.e., additional entities and attributes that are not populated for the data exchange).

Every AIM entity should be traceable back to an ARM application object (AO) or AO attribute. It is advantageous to verify this during the consistency checks that are discussed earlier in this document. If an AIM entity is traceable to more than one ARM AO or AO attribute, then the mapping needs to address this inconsistency and correct the mapping table.

Notes to consider working in:

During interpretation process lots of ARM questions are asked and clarifications come up in the discussions, where are they ever documented besides in the AIM? It is impossible to trace back from the AIM to a requirement that was never there in the first place. Minimum, should be documented in the interpretation report? Important things -- AP team is asked to go back and clarify the definitions in clause 4. Units is an example. Do not add application objects, just clarify definitions in clause 4.

global_units sometimes does not get brought in explicitly because ARM is not sufficiently detailed. This should go into ARM some how or go into mapping rules etc.

Recommendation for incorporating items from mapping table in tothe documentation of the AIM, could require enumerated values of the strings to be included in the imported entity modifications or local entity definitions. There is a problem with some implementors looking only at the ARM and the AIM, though they need to look at the entire AP, it would help to have as much information as possible in the AIM definitions.

Should application_protocol_definition be reflected in the mapping table? This is not so much an application requirement, but a ISO 10303 requirement. 214 has included it in a mapping rule for an entity at the product/item level.

You get mapped items implicitly via the AICs, what if they do not want to instantiate mapped_item ever? should they write a rule that says there are not instantiable? This has not been done in other APs. Mapped_item is explicitly interfaced into the AIC so is, in theory, instantiable in the AIM. Hmmm. Traceability issue. You will have this extra subtype of rep_item that is mapped_item. You will have more than you had planed in your select lists and subtypes after pruning.

8.2 Building consistency of interpretation across APs

AP planning projects should establish a planning model for the domain and clearly define scope boundaries and overlaps of known and planned APs. These planning models should be explained at WG3 and WG12 plenaries to ensure broad understanding of each AP project's scope.

AP projects, in collaboration with the relevant AP planning project, should work to identify common concepts across APs, AP unique concepts, and central concepts for each AP. This should include identifying similar UoFs across the APs and requirements for inteface conformance classes across APs. AP project shall review UoFs of other APs and AP projects for overlapping concepts and UoFs. When overlaps are identified, these conclusions shall be included in the interpretation report and a copy of this analysis shall be forwarded to WG12.

AP planning projects shall maintain a glossary of common terms and concepts shared across related APs. As the APs are developed, the terms and concepts will most likely be refined. The related AP projects should endeavor to maintain this consistency in the usage of these terms and concepts.

For AP information requirements that are the same, the objective of ISO 10303 is to ensure a consistent representation of that information in all APs. The assessment of the commonality and uniqueness of information requirements from different APs requires careful analysis and the participation of domain experts from the relevant AP projects. Based on the result of this analysis and the market demand for interoperability between implementations of the APs, AP projects should consider:

— generalizing requirements to make them the same in order to enable the use of common AIM structures across the relevant APs;

— for information requirements which are the same, using the exact same AIM structures and entity names;

— for information requirements which are similar but not the same, documenting the unique characteristics and how common AIM structures with some additional structures or different values are used to represent the different variations on the common requirements.

The results of this work are included in the AP Interpretation Report and are presented to WG12 as input to WG12's UoF/AIC management log.

WG12 shall provide templates for consistent use of IR constructs, e.g., product_definition_formation for AP 216 vs. AP 214

The Quality Committee shall establish and maintain a standard set of values, e.g., unit and measures values, roles for asssignment entities, and application context names, for use in APs.

Quality Committee. or WG12 should develop naming conventions for AIM constructs, e.g, subtype names, organization_assignments, and guidelines for consistency of AIM structures, e.g., group_assignment and select types names for approval.

Select types for management resources:

— group_assigned_item the select type that was created for 216 is intended to be used for all other ship APs.

— grouped_item vs. auto_design_grouped_item Interpretation has flip flopped on this guidance.

— Select type itself has the same semantics in each AP, it is a grouping for selection. (MG point)

— There are some cases where the semantics are different, and some cases where the semantics are the same. (YY and AW agree on this point)

— If there are general semantics associated with "standard" names for select types for the management resources, those need to be documented somewhere so they can be used by all APs. Use same name for same semantics

Subtypes for the management resources

— Same or different names per AP? If conflicting semantics as shown by local rules, then they should have different names. If no rules, then agreement could be reached and the same name used. YY thinks subtype should be unique to the AP, select types can be shared. (She said subtypes have different semantics because they select different things. I think in that case, the select types are more different and less shareable than the subtypes.)

The management resources can be standardized across APs for subtypes. For example ship_building_group_assignment, automotive_group_assignment. The select type name can be standardized and interpretation is willing to follow if someone lists select types.

Each AP likely to have one organization_assignment per AP named <blah>_organization_assignment, Someone needs to come up with a set of allowed <blah> and document it in the Interp guidelines for AP interoperability section.

Local rules in subtypes are added to be precise in specification of APs requirements in an AIM. In some cases the decision may be made to not put local rules in the subtype to allow sharing of the entity definition across other APs. There is a tradeoff between shareability and precision. It would be nice to have some guidance on what the tradeoffs are and decide whether they want to be precise or share with other APs. One extreme is to remove all rules and add everything into the select type so it is completely flexible. Shareable portion can only accommodate what is known today. Do what you know you can do today. Plan ahead, harmonize requirements with other APs you intend to share data with. This is not difficult to look through the AP and see what you need to assign approvals to, etc.

When to create a new subtype ofa management entity and when to re-use an existing one. What is an exaple where you need to create a second subtype per an AP? Mitch says when the role is added, then will be able to get away with only one subtype per AP.

New semantic and completely different set of management resourses assignment assignments are always the same. Semantics of what being assigned is different. Group assignment in particular has different usages/semantics. In order to capture this, different subtypes are needed.

At beginning of week we were always mapping to product_definition but not product. Why did ship map to product definition not product? It always has a definition

If the mapping of an application object does not always require property associated with it, it is always mapped to product. When it does, it maps to product_definition. No property associated with equipmentin 227 so mapped to product. When maped to product definition, most always has mapping rule that ties it up to product. Product_definition is also mapped to when there is lifecycle info also. When you map to product_definition the structure demands that there is a product. Although you always have product in scope, even if only product_definition usually something maps to product.name or product.id.

In 216 only ship maps to product_definition.

With measure_with_unit use of pairing of measure_ith_eunit. value _component and unit_component should have a standard set established for all APs

roles for assignment entities
application_context_element.name for all 4 types of contexts

etc. general issue of needeing to standardize on these strings. Get PDES, Inc. AP Interop team to propose a baseline set from existing APs.

we defined table that is not restricted to the use in the AIC. This could happen in other situations. When they encountered tables in 214 for the first time they did not think ti would become a resource, or an AIC.

Annex A

Commonly modeled concepts

This annex consists of a table that contains concepts that are commonly modeled in application reference models of ISO 10303 application protocols. The concepts may be given slightly different semantics in different APs. These concepts may also be modeled differently in different APs. The different models are provided as are common interpretation results.

(awaiting EXPRESS-G diagrams from Rob Howard/216 team)

Concept	AIM Structure	What it means
AGGREGATION(collection) 1) aggregation of propertes for a lifecycle view (see diagrams) 2) elements of a technical description (items of a representation 3) administrative information	1) property_definition.definition -> product_definition (a. existence dependency and multiple instances) property_definition_representation .definition -> property_definition (b. intersection entities allowing 2 way aggregation resolving many to many relationship (independent)) (c. relationship entity and things being aggregated --example class_ntation: hull notations and machinery notations use of peer relationships) (see picture back of p.5) 2) representation.items -> rep_item (d. set of items on a representation) 3) management resources templates/ assignment constructs	1) aggregation of properties for a property definition is very high level de products of property: material, shape, s There is a set of classes of property tha rep. Any property that you want to say A cup has a shape (that is a property_de height (height is part of the shape, so is aggregation of high level propertyies th product_def and also the aggregation of representations of the shape property_d to representation 2) name-value pair 3) template used in AP to provide other entity--the half in the IR points to the th being assigned to. This hald is a clised concepts in the IRs. The other end is n the distinction in the ARM Distinction b as oposed to administrative or managme Owner is not a property of something. owner in representation structure

Concept	AIM Structure	What it means
COMPOSITION(whole/part) 1) Product/Part Product (see notes for possible modeling structures)	1) Product_definition/ Product_definition_relationship relationship	1) the composition will be lifecycle dep how many lifecycle compositions
2) Shape/Part Shape	2) shape_aspect.of_shape-> product_definition_shape	2) shape aspect is existence dependent c part/product.
3) Representation/Part Representation	3a) representation_relationship.relating -> and .related -> representation	3a) rep_rel is a loose relationship relating is the result of the relationship other where you're just asserting they a be a transformation specified (has subtypes: rep_relationship_with_tr more mathematical information about th presentation_rep_relationship constrains and how transformation can be specifie
	3b) mapped_item	3b) mapped_item takes one rep and mal another rep. Always a transformation ir Symbol instance on drawing is rep item the drawing, that is accomplished with r
4) Property/Part Property	4) property_definition_relationship and property_definition	4) same as this is a template for all types of relatio specific existence dependencies that hav

Concept	AIM Structure	What it means
SUCCESSION (order) (document in ARM via list attribute or via preceding/succeeding)	1) xxx <- xxx_relationship.related and <- xxx_relationship.relating 2) representation_item.name carries sequence id representation.items S[1:?] -> representation_items	1) using relationship entities to say ther predecessor in the list. Two ways of do the standard data values for the chain of succession? related is the succeeding and relating is 2) sequence_id used when there are spe representation_item (because no rep_ite stuck with set relationship between rep Red_item.name carries sequence id
CLASSIFICATION (categorization) grouping for classification purposes 1) of products 2) elements of product_data and other stuff (not product, really)	1) product_category - is the class product_related_product_category.products -> product - is the classification there is a relationship entity on product_category that can be used to allow for composition of categories 2) group/group_assignment structure	1) requirements for general categorisati categories (paricularly when ARM cont tree or IDEF1X categorization) Classifi than Usage: could use as is in resources (Example ship_designation.ship_type m product_category.name) 2) classification of attributes of the thin about Usage: have to subtype group_assignme also subtype group. Don't subtype grou and only one usage of group in the AP. entity modification to explain usage) Example: class notation

Concept	AIM Structure	What it means
GROUPING grouping things for a purpose, a shared connotation, Example: groups of representation_items groups of product_definitions groups of things of dissimilar types	group_assignment structure from integrated resources. group is the collector the collection is the group/group_assignment and the set of items in the assignment	connotation captured by name description gets description of conotatio and group_assignment.role.name identi assignment--could be why the group is collection of items VERY GENERAL
CONNECTION 1) A is connected to B 2) A is connected to B using C 3) A has connecting bit that connects to the connecting bit of B Note: All of above can be logical and physical. Not a whole/part relationship	1a) product_definition <- product_definition_relationship.relating, .related 1b) shape_aspect <- shape_aspect_relationship.relating, .related 2) subtype of product_definition and product_definition_relationship 1b) shape aspect <- shape_aspet_relationship	1a) product definition -- pipe connectin connection to terminal 1b) shape aspect_relationship connectio the collection is a product definition. C the related in a product_definition. Ass the things being connected, the thing m 1b) shape aspect related to the shape of components. OR the shape aspect of the aspect of the thing threaded. OR the sha relationship. Note: Makes difference to contexts of t which the stuff is connected. (Can't ha aspect) (Logical points to product_defin either shape or product_definition)

Concept	AIM Structure	What it means
IDENTIFICATION 1) primary identifier a) if appropriate attrribute in integrated resource b) if not 2) aliasing (primary and aliases)	 a) natural attribute in integrated resource construct--use that instead b) identification_assignment template	 a) If there is an attribute id (or possibly construct mapped to and there is only o always map to that if you can. b) if requirement is to identify somethi different languages or by different orga or if the
NAMING always human interpretable 1) giving something a name regardless of semantics--any instance of any data type Only semantic is name 2) if ARM has name attribute and	 1) name_assignment template 2) .name of IR construct	 1) Rarely used, example 3 instances of name_assignment. Not same as identifi cad data/geometry in 204, 205 and 221 anything anywhere. Claim they have re arbitrarily name All three cases it is pro mechanism for naming instances indepe type. 2) standard data convention-- use .name for relationship entities to ca from the AP. If not relationship entity, use .descriptio

Annex B

Notes to consider

Explain the difference in interpretation practice for one focused AP versus for one AP being within a suite of APs:

- usual interpretation practice is to map an ABS supertype to an OR case of the specific things (subtypes) for single; map to the more generic thing for suite AP with mapping rule that restricts the mapping for the particular path
- naming of assignment entities
- mapping to more generic constructs
- use fewer local rules? Mitch disagreed with that comment

For free stuff:
configuration_item has 2 required attributes. how would this be instantiated? 1st case the information is there but not pertinent--ignored. 2nd case, information was omitted, and value can be readily found.
In thefirst case tou cannot find out the value. YY always recommends if an ID put in some kind of simple id cd1 cd2. If description, put n/a, not specified or something. She highly recomends not doing blank or null. At some point in time, you may want to search on those strings. If everything is blank, you can't distinguish between them. Document strategy for populating these strings in the Usage and Implementation Guidance annex of the AP. (Recommended Practices). Problem is that this guidance can change. Should it really be standardized? or should it go into a separate document.

IRs are modeled using principles of existence dependence. The AP ARMs are often modeled using principles of definition_dependence. There needs to be a discussion of the differences and why this often results in changes to the ARM.

capture rationale for .name .description
.description states what the particular instance is used for and
.name says what the type name is

Useful to use instance diagrams to document actual constraints and illustrate paths. For less experienced interp resources, recommend consistent use of instance diagrams

Grey areas in use of mapping rules between assertions and
if there is an assertion that defines the path, and there is a mapping rule that applies to the path, the rule needs to be documented in the assertion, not the entity
*** come up with more explanations for all situations ***

is the mlt supertype for semantic or structural reasons? What does it me an to be a rep and a rep_item? Semantically, it is a colection of items that are used also as an element in the representation fo something else. Mapped item could be used for that but implies a transformation from one context to another, an there is no such in this. Table representation subtype of symbol rep in part 46 is a mapped item, that is accurate as there is a transformation in it.

At the resource level, we can create a relationship between rep_items. Can't do that in the AIM. This is probably something that should be an extension to part 43. What would be the extension to part 43 -- include composite rep_item that has a list or set of rep_items. Or to add a rep_item_relationship. Rep_item_relationship would have an impact, not just be an addition.

Class designation. 2 mandatory relationships. 2 uses for the AND in the AIM element column: one where there are two specific instances required to meet the mapping. The other is when there is a complex instance required. These two cases are not differentiated in the syntax. In the case where the two instances are type compatible, it would be difficult to tell what was intended, both instances or a complex instance.
We are assuming for this workshop that the management resources are made consistent, all have .role, etc.
These are proposed solutions to the SEDS

Why is one mapped to a subtype of rep and the other mapped to rep.
hydrostatics_definition mapped to subtype of rep -- has whole boatload of additional constraints that must be accommodated by the creation of a subtype
moulded_form_design_definition maps directly to rep

Tuesday Questions and answers:
Clarifications of action schema needed. Some other problem due to the approval schema in the ARM being unclear to current 216 resources. In this location consistency checking need to review whether the interp is adding tendencies that are not intended by the ARM.

map the concepts before picking the EXPRESS
predecessor/successor relationship ??? Julian

having selected a construct, determining whether constraints are necessary
The guidance of creating as few subtypes as possible is interesting, as it would make us want to revisit interps of early aps.
What are the precise criteria for determining whether a subtype shall be created? How many constraints?
Usage of attributes in the IR construct--if there is an attribute that can hold the constraint, then no subtype
1)More to do with behaviour or concepts related to it. Approval_history -- some kinds of approvals it is composed of, but they are not approval historts. Some required associations, required attributes. Same kind of thing used differently in requirements.
2) Different constraints on same IR construct--need subtype for each different set of constraints. Define a scope for the constraints. Representation item is one that gets subtyped alot

In which situation are you defining a subtype that is different from the parent, or just has different population cause there is no other way to descriminate. that is why role is so important. With assignment entities, without roles, must create different subtype with different name. That is why roles are being added to the assignment entities.

Use of mapping rules. Constraints (may) not be specified in the EXPRESS. Mapping rules typically only pertain to one mapping, not to the entire set of populations of something. Not constraining total population set across the AP. Not imposing constraint on definition of a type, just on the populations of that type.
Change to AP documentation process!!! ->

If AIM created subtype or IR imported entity modifications--the constraint in the mapping rule should be documented in the standard data values for approval_status.name (there should also be a global rule that states all allowable values cause there is a bounded population for a given attribute)

Global Rule: (there should also be a global rule that states all allowable standard data values cause there is a bounded population for a given attribute)

lots of add'l attributes, how do they get populated? Traceability discussion from last week.

Proposed changes to the interpretation
(all roles we mapped to except for person role , organisation role, date_time role)

Initial set of IRs responded to requirements form the initial APs (41-44)
Next set of IRs were used as place for requirements from subsequent APs (45, 47, 49) now they are closed
Now we are in the position of not having any place to stick extensions to the IRs. We need to get a NWI opened as a home for the SEDS resolutions and extensions.

TYPES OF RELATIONSHIP (how to determine relating/related)
related thing is always the dependent thing in the relationship
relating is always the independent
if the arm has preceding and succeeding, preceding is always independent (relating)
preceding/succeeding -> relating/related
whole/part -> relating/related
peer/peer collection relationship collector/collected relating/related
class_notation is collector that collects _ and _ notation
class notation is always relating
machinery_notation and hull notation are always the related

Jochen suggested that we could have a table where general concepts that can be easily identified and show options in IRs to represent these concepts.

One common one is usage of list in EXPRESS--requirement for successor/predecessor
when do you use a relationship entity in the AIm where thereisn't one in the ARM.

Annex B

Bibliography

1. ISO 10303-41:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 41: Integrated generic resources: Fundamentals of product description and support.*

2. ISO 10303-42:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 42: Integrated generic resources: Geometric and topological representation.*

3. ISO 10303-43:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 43: Integrated generic resources: Representation structures.*

4. ISO 10303-203:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 203: Application protocol: Configuration controlled 3D designs of mechanical parts and assemblies.*

5. ISO 10303-216, *Industrial automation systems and integration — Product data representation and exchange — Part 216: Application protocol: Ship moulded forms.*